BeagleBone Cookbook Webinar Series
Recipe #2
Displaying GPIO status in a Web Browser

November 3, 2015

Jason Kridner

Co-author of BeagleBone Cookbook

Board member at BeagleBoard.org Foundation

Sitara Applications Engineering at Texas Instruments

BeagleBone Black

Ready to explore and use in minutes

Truly flexible open hardware and software development platform

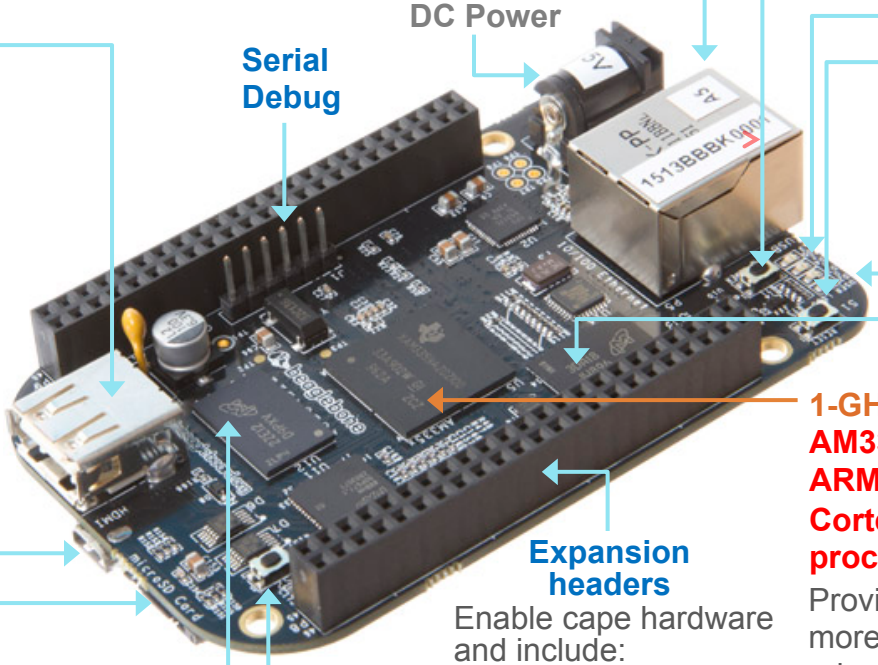
All you need is in the box

Proven ecosystem from prototype to product

- Ready to use
  - USB client network
  - Built-in tutorials
  - Browser based IDE
  - Flashed w/Debian
- Fast and flexible
  - 1-GHz Sitara ARM
  - 2x200-MHz PRUs
  - 512-MB DDR3
  - On-board HDMI
  - 65 digital I/O
  - 7 analog inputs
- Support for numerous Cape plug-in boards
  http://beaglebonecapes.com

BeagleBone Black – the most flexible solution in open-source computing

BeagleBone Black board features



10/100 Ethernet

USB Host
Easily connects to almost any everyday device such as mouse or keyboard

microHDMI
Connect directly to monitors and TVs

microSD
Expansion slot for additional storage

512MB DDR3
Faster, lower power RAM for enhanced user-friendly experience

Serial Debug

DC Power

Boot Button

Power Button

LEDS

Reset Button

USB Client
Development interface and directly powers board from PC

4-GB on-board storage using eMMC
- Pre-loaded with Debian Linux Distribution
- 8-bit bus accelerates performance
- Frees the microSD slot to be used for additional storage for a less expensive solution than SD cards

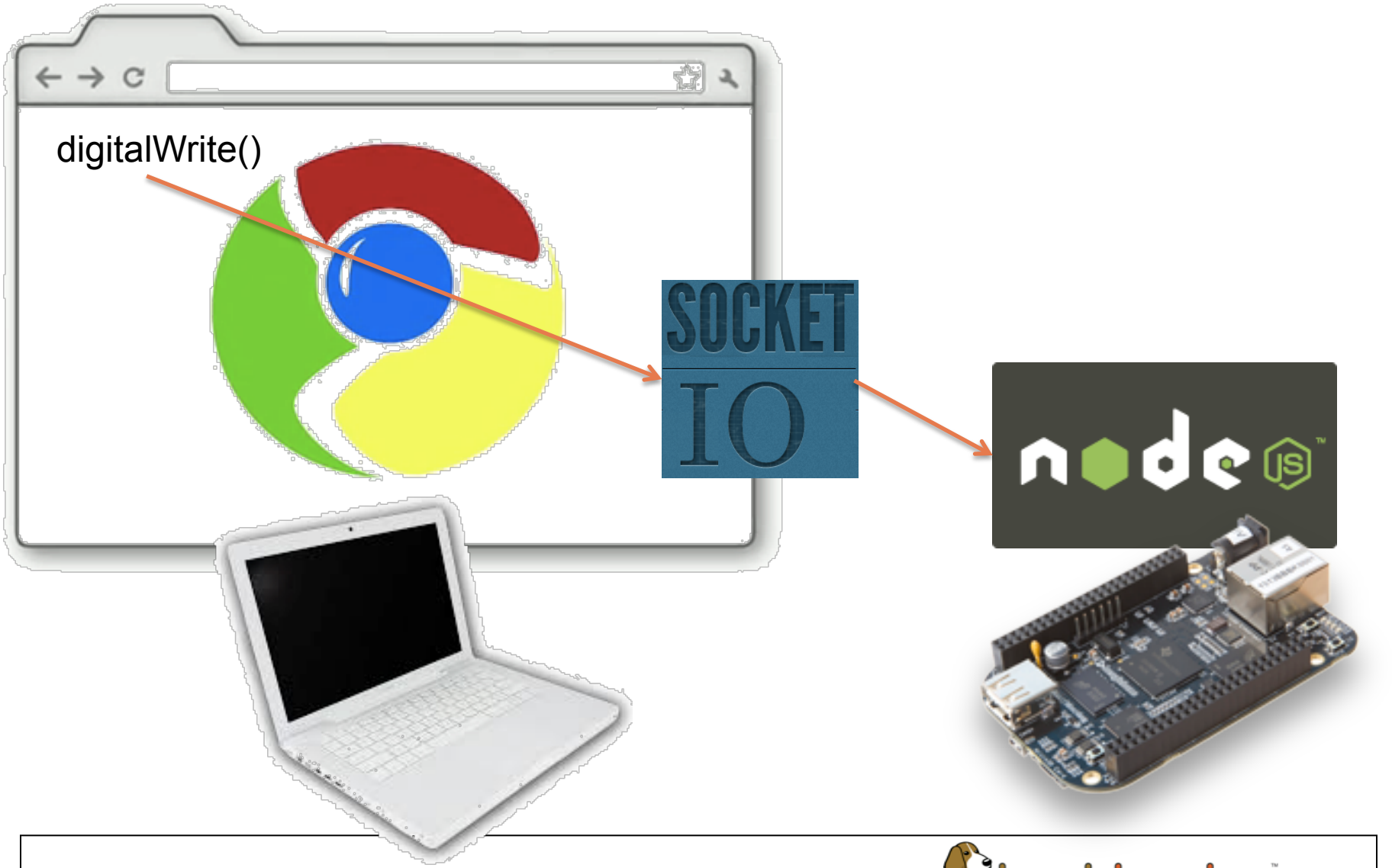
1-GHz Sitara AM335x ARM® Cortex™-A8 processor
Provides a more advanced user interface and up to 150% better performance than ARM11

Expansion headers
Enable cape hardware and include:
- 65 digital I/O
- 7 analog
- 4 serial
- 2 SPI
- 2 I2C
- 8 PWMs
- 4 timers
- And much much more!

Money saving extras:
- Power over USB
- Included USB cable
- 4-GB on-board storage
- Built-in PRU microcontrollers

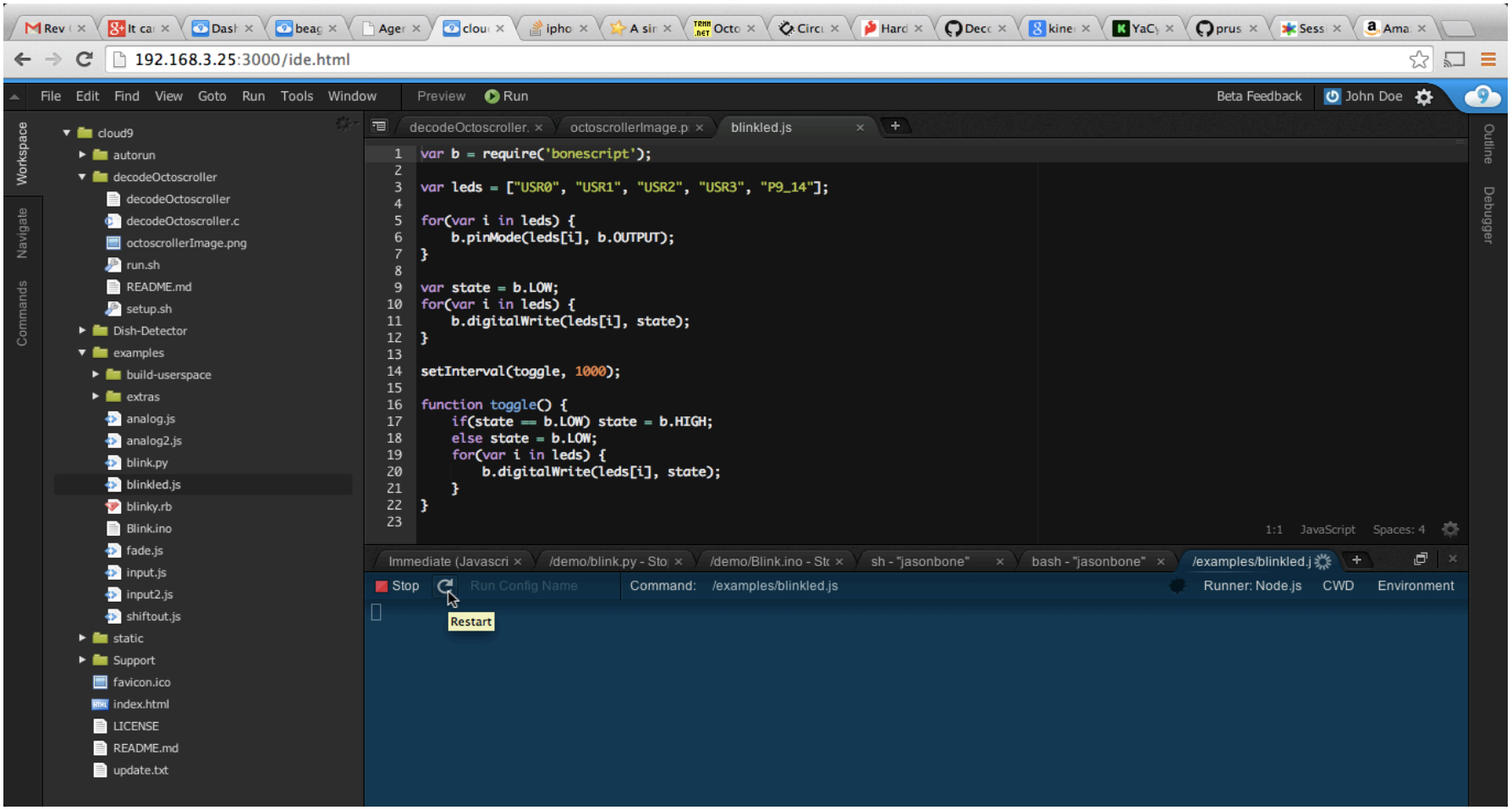
Simple browser-based interactions

http://beagleboard.github.io/bone101

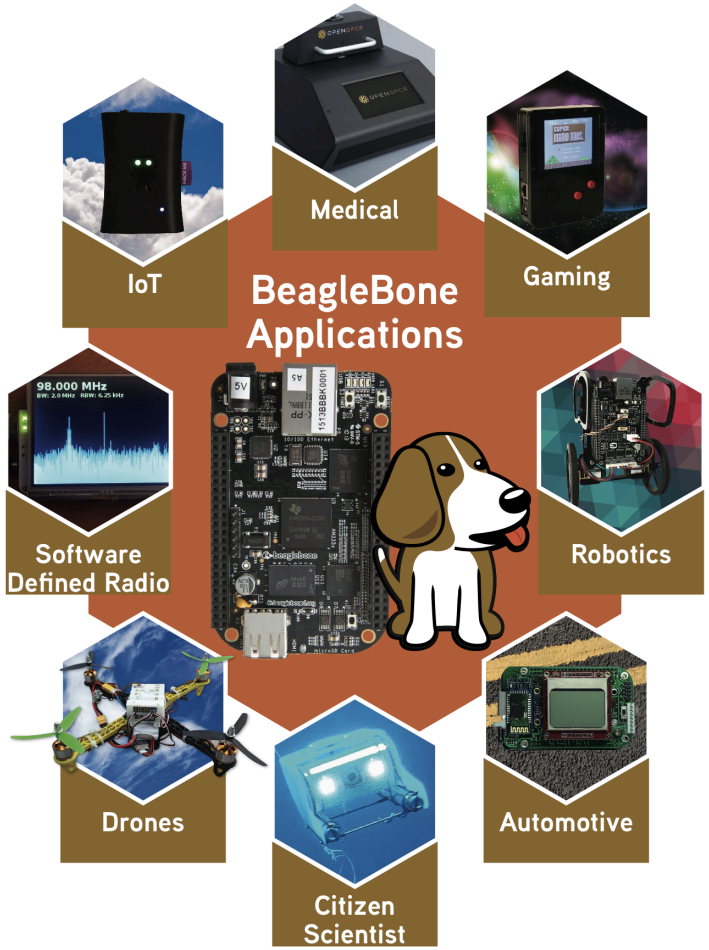


Cloud9 IDE hosted locally

Zero install and exposes command-line



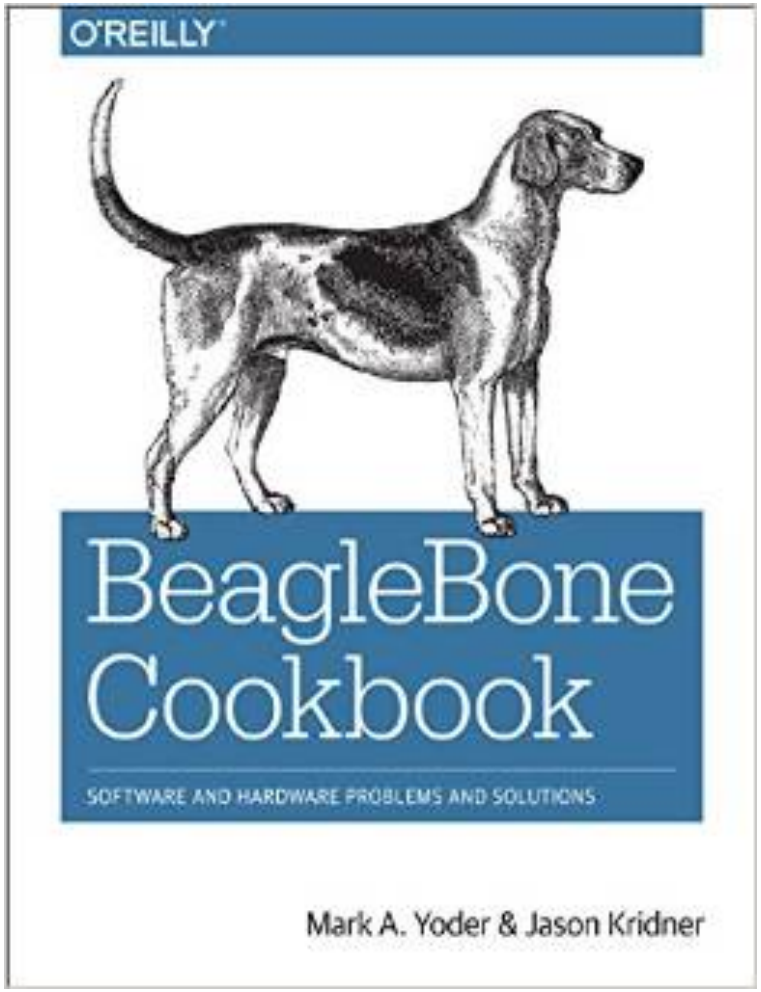
10,000s of developers building connected devices today



- Medical analysis, assistance and information management
- Home information, automation and security systems
- Home and mobile entertainment and educational systems
- New types of communications systems
- Personal robotic devices for cleaning, upkeep and manufacturing
- Remote presence and monitoring
- Automotive information management and control systems
- Personal environmental exploration and monitoring

BeagleBone Cookbook

http://beagleboard.org/cookbook



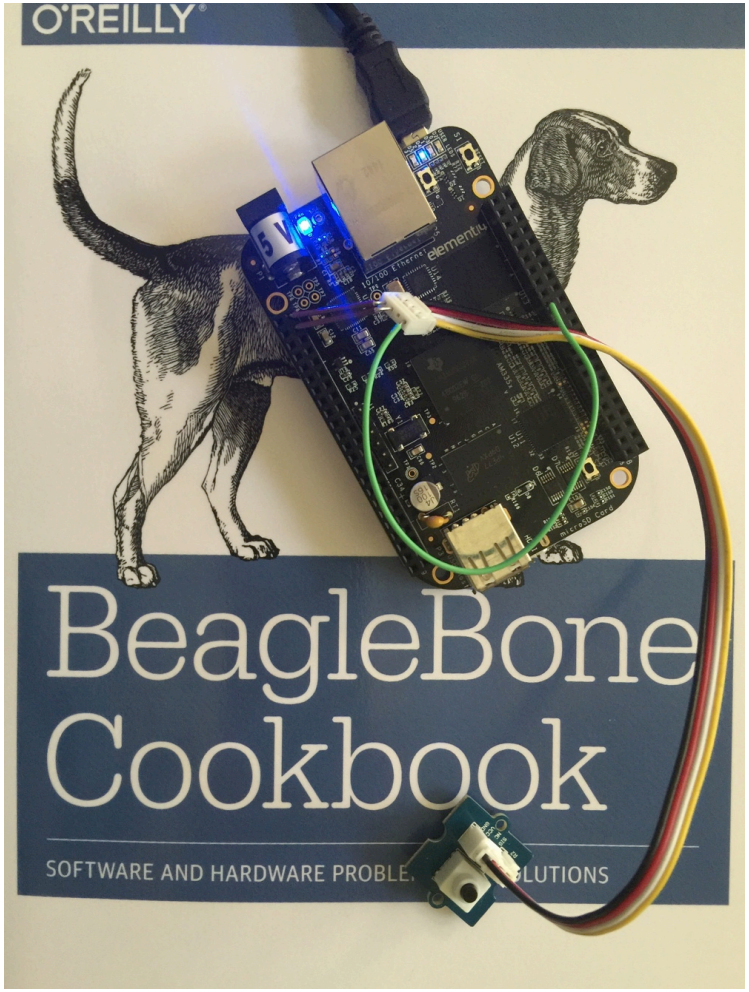
- 99 recipes covering
  - Basics
  - Sensors
  - Displays and outputs
  - Motors
  - Internet of things
  - Kernel
  - Real-time I/O
  - Capes

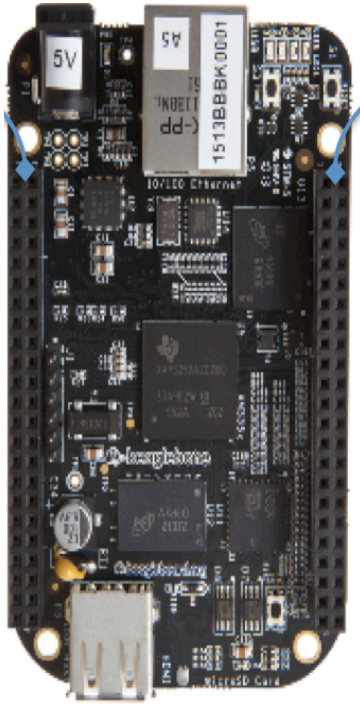
Prerequisites

- Connect to the board per recipe 1.2
  - http://beagleboard.org/getting-started
- Verify the software image per recipe 1.3 and potentially updating per recipe 1.9
  - http://beagleboard.org/latest-images

Connect a button to GPIO P8_19

http://beagleboard.org/Support/bone101/#headers





LEGEND
Power/Ground/Reset
Available Digital
Available PWM
Shared I2C Bus
Reconfigurable Digital
Analog Inputs (1.8V)

P8

DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

Recipe 6.6: Continuously Displaying the GPIO Value

https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/jQueryDemo.html

```
<html>
<head>
  <title>BoneScript jQuery Demo</title>
  <script src="/static/jquery.js"></script>
  <script src="/static/bonescript.js"></script>
  <script src="jQueryDemo.js"></script>
</head>

<body>
<h1>BoneScript jQuery Demo</h1>
<p>buttonStatus = <span id="buttonStatus">-
</span>
</p>
</body>
</html>
```

https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/jQueryDemo.js

```
setTargetAddress('192.168.7.2',
    {initialized: run}
);
function run() {
    var b = require('bonescript');
    b.pinMode('P8_19', b.INPUT);
    getButtonStatus();
    function getButtonStatus() {
        b.digitalRead('P8_19', onButtonRead);
    }
    function onButtonRead(x) {
        $('#buttonStatus').html(x.value);
        setTimeout(getButtonStatus, 20);
    }
}
```

Stepping back to recipe 6.3

Interacting with the Bone via a Web Browser

https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/server.js

```
var port=9090, h=require('http'),
    u=require('url'), f=require('fs');
var s=h.createServer(servePage);
s.listen(port);

function servePage(req, res) {
 var p = u.parse(req.url).pathname;
 f.readFile(__dirname+p,
  function (err, data) {
   if (err) return;
   res.write(data, 'utf8');
   res.end();
  }
 );
}
```

- BeagleBone Black ships with Debian and Node.JS
- Using Node.JS is easy to serve up a simple web page
- Run with:
  node server.js
- Browse to port 9090 and a local file

Recipe 6.4 adds hardware interaction

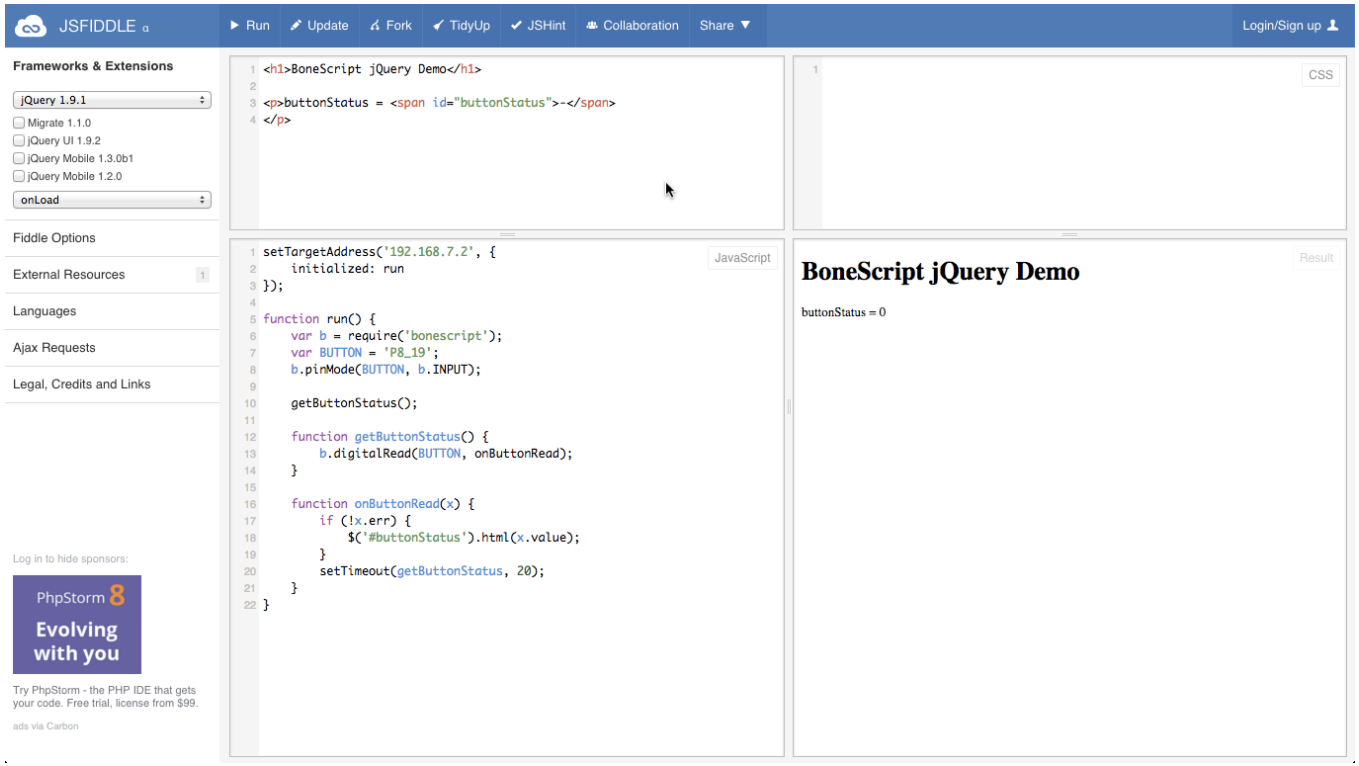
https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/GPIOserver.js

```
var h=require('http'),f=require('fs'),
    b=require('bonescript'),
    g='P8_19', p=9090;

var htmlStart = "<!DOCTYPE html>\
<html><body><h1>" + g + "</h1>data = ";
var htmlEnd = "</body></html>";
var s = h.createServer(servePage);

b.pinMode(g, b.INPUT);
s.listen(p);

function servePage(req, res) {
    var data = b.digitalRead(g);
    res.write(htmlStart + data + htmlEnd, 'utf8');
    res.end();
}
```

- Builds on simple Node.JS web server
- BoneScript library utilized on server
- Content served using variables, not files
- Full example uses URL path
  - distinguish content
- Refresh manually

Recipe 6.5 introduces jQuery

http://jsfiddle.net/n5j3p32o/1/



- Great tool to make content dynamic
- jsfiddle.net provides a playground for learning
- Learn more about the API at jquery.com

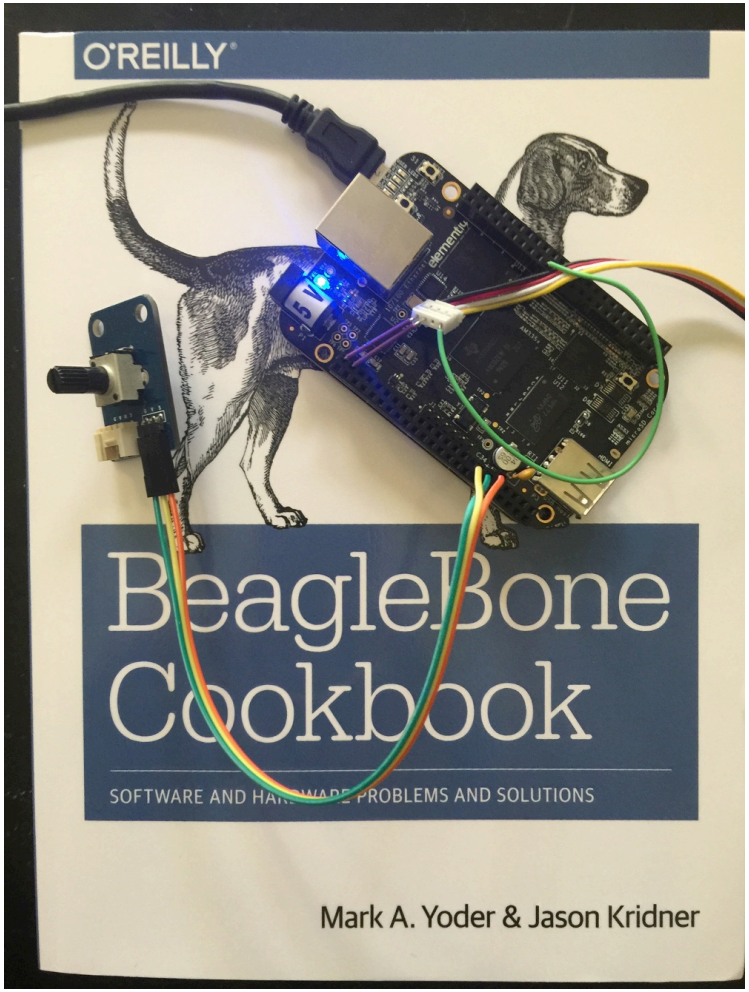
How BoneScript works in the browser

http://beagleboard.org/static/bonescript.js

- Provides a setTargetAddress() function to define the global require() function
- Utilizes the built-in Node.JS based web server built into the BeagleBone Black default image https://github.com/jadonk/bonescript/blob/master/src/server.js
- On-board bonescript.js provides the require() function and utilizes socket.io to define remote procedure calls https://github.com/jadonk/bonescript/blob/master/src/bonescript.js

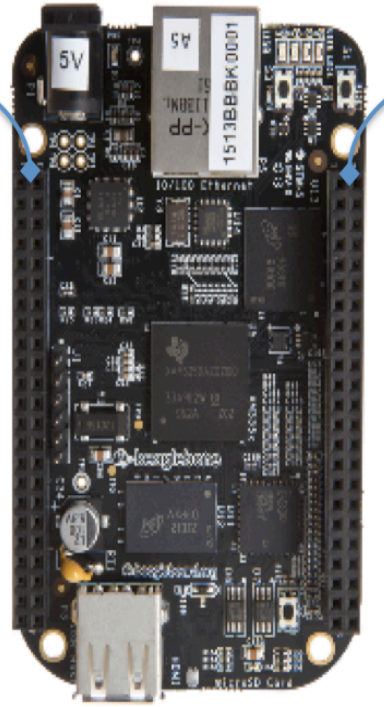
Connect a potetiometer to ADC P9_36

http://beagleboard.org/Support/bone101/#headers



P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPI0_CS0	17	18	SPI0_D1
I2C2_SCL	19	20	I2C2_SDA
SPI0_D0	21	22	SPI0_SCLK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_112
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND



LEGEND
Power/Ground/Reset
Available Digital
Available PWM
Shared I2C Bus
Reconfigurable Digital
Analog Inputs (1.8V)

Recipe 6.7: Plotting Data

- See demo code at
  - https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/flotDemo.js
  - https://github.com/BeagleBoneCookbook/firstEdition/blob/master/06iot/flotDemo.html
- This is just the beginning
  - Lots of different types of hardware interactions
  - Lots of different visualizations possible in the browser

More

- JavaScript tricks
  - http://beagleboard.org/project/javascript-tricks/
- Shortcuts to updates and examples from the book
  - http://beagleboard.org/cookbook